**DL-HW #10**

2015004693\_양상헌

*실행환경: MAC OS TERMIINAL( MACBOOK PRO 2015 RETINA , MOJAVE 10.14.6 ) , ANACONDA*

1. **Source Code:**

* *Assignment10 폴더 참조*

*(DL\_HW\_10week\_basicRNN.py, DL\_HW\_10week\_LSTM.py, DL\_HW\_10week\_GRU.py)*

스크린샷, 전화, 테이블, 검은색이(가) 표시된 사진

자동 생성된 설명

9주차 실습 과제물의 일부분을 조금씩 다르게 설정하여 과제를 진행하였다. Sample sentence가 더 긴 문장으로 바뀌었고, 기존의 hidden\_size를 15로 고정시켜 주었다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

BasicRNN, LSTM, GRU 세가지 모델의 성능차이를 비교해보기 위해 소스코드를 3개로 분리하여 실행해보았다. ( “DL\_HW\_10week\_XXXX.py” 에서 XXXX 부분이 사용한 모델을 의미한다. )

1. **Result:**

3가지 모델(BasicRNN , LSTM, GRU)를 각각 연속적으로 10회씩 실행시켜 결과를 캡쳐 하였다. 각각의 결과 캡쳐 사진은 assignment10/result\_of\_XXXX 폴더에 정리해 놓았고 XXXX가 사용한 모델의 종류를 의미한다.

* **BasicRNN(*DL\_HW\_10week\_basicRNN.py* ) 실행결과**
* Minimum Loss & Correct Sentence (Best Result):

창문, 건물, 앉아있는, 하얀색이(가) 표시된 사진

자동 생성된 설명

* Maximum Loss & Wrong Sentence (Worst Result):

하얀색이(가) 표시된 사진

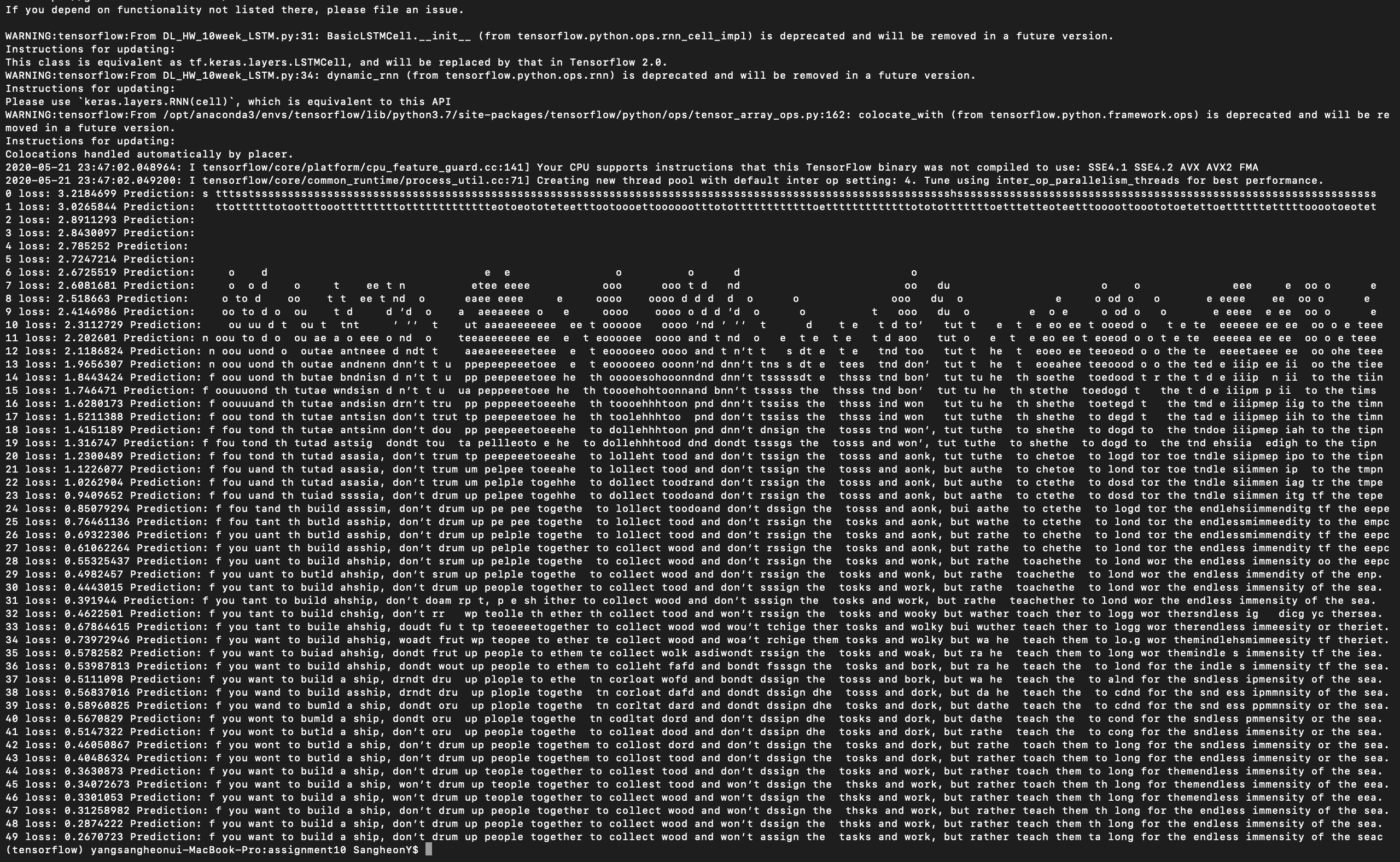
자동 생성된 설명

* **LSTM(*DL\_HW\_10week\_LSTM.py* ) 실행결과**
* Minimum Loss & Correct Sentence (Best Result):

앉아있는, 검은색, 하얀색, 물이(가) 표시된 사진

자동 생성된 설명

* Maximum Loss & Wrong Sentence (Worst Result):



* **GRU(*DL\_HW\_10week\_GRU.py* ) 실행결과**
* Minimum Loss & Correct Sentence (Best Result):

창문, 건물, 하얀색, 남자이(가) 표시된 사진

자동 생성된 설명

* Maximum Loss & Wrong Sentence (Worst Result):

앉아있는, 하얀색, 물이(가) 표시된 사진

자동 생성된 설명

1. **Disscussion:**

**< 코드 설명 >**

**스크린샷, 전화, 테이블, 검은색이(가) 표시된 사진

자동 생성된 설명**

스크린샷이(가) 표시된 사진

자동 생성된 설명

소스코드의 전체적인 구조는 9주차 과제의 구조와 같다. 다른 점은 Sample Sentence가 훨씬더 길어진 점과, hidden\_size를 15로 고정하여 사용한 점, 그리고 3종류의 RNN 모델을 각각 따로 사용한 점이다.

**< 결과 분석 >**

3가지 모델을 각각 따로 실행 시켰을 때 큰 성능 차이가 있을 것이라고 예상했지만, 생각보다 그렇게 큰 성능 차이가 나지는 않는 것 같았다.

단, 각 모델의 결과에서의 Loss 의 크기와 결과 문장의 정확성의 관계가 각 모델마다 조금씩 차이가 있는 것을 확인할 수 있었는데, LSTM이나 GRU의 경우 상대적으로 basicRNN보다 더 큰 Loss 값을 가져도 결과 문장이 더 올바른 문장에 가깝게 되는 경향을 확인할 수 있었다. 문장의 완성도 혹은 정확성만 놓고 비교해본다면 GRU > LSTM > basic RNN 순서의 성능을 가진 것 같다.( 실행 횟수의 표본이 작기 때문에 올바른 분석이 아닐 수 있지만, 각각 모델들을 같은 횟수 만큼 실행해 봤을 때 이러한 경향이 있음을 확인했다. )

또한 epoch을 50으로 제한하고 실행한 부분도 큰 성능차이를 보이지 못한 원인으로 볼 수 있을것 같다. Epoch 숫자가 작아서 실제로 더 최적화가 될 수 있는 상태이지만 일정 횟수가 끝나면 바로 학습을 종료해버리고 결과를 반환하게 되어 있어서 원하는 대로 결과가 잘 나오지 않은 것 같기도 하다. Epoch 숫자를 조금만 더 조정해보면 각각의 모델의 성능 차이를 더 뚜렷하게 확인 할 수 있을 것 같다.